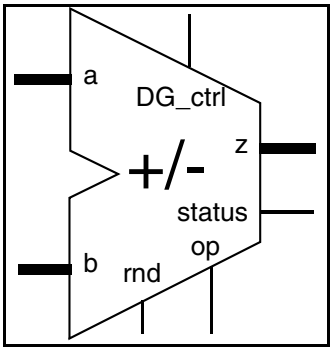
# DW_fp_addsub_DG

## Floating-Point Adder/Subtractor with Datapath Gating

Version, STAR, and myDesignWare Subscriptions: IP Directory

## Features and Benefits

Revision History




- Has the same functionality as DW_fp_addsub when the component is in normal operation
- Consumes less dynamic power than DW_fp_addsub when disabled (inputs are active, but the output is not being used)
- Precision is controlled by parameters and covers standard IEEE formats
- Exponents can range from 3 to 31 bits
- Fractional part of the floating-point number can range from 2 to 253 bits
- Configurable to be fully compliant with the IEEE Std 754-1985 standard
- Configurable for NaN representation compatible with the IEEE Std 754-2008 standard (controlled by the *ieee_compliance* parameter)
- DesignWare datapath generators are employed for better power and QoR

## Description

DW_fp_addsub_DG is a floating-point component that adds or subtracts two floating-point values, `a` and `b`, to produce a floating-point result `z`. The control over the arithmetic operation is through the input `op`. Also, a control input (`DG_ctrl`) can disable the component to reduce dynamic power consumption when the component is not in use and inputs are still active.

Component pins are described in Table 1-1 and configuration parameters are described in Table 1-2.

**Table 1-1 Pin Description**

| Pin Name | Width | Direction | Function |
|---|---|---|---|
| a | (*sig_width* +*exp_width* +1) bits | Input | Input data |
| b | (*sig_width* +*exp_width* +1) bits | Input | Input data |
| op | 1 bit | Input | Defines the operation:<br>■ 0: Addition<br>■ 1: Subtraction |
| z | (*sig_width* +*exp_width* +1) bits | Output | a `op` b |
| status | 8 bits | Output | Status flags for the result<br>For details, see STATUS Flags in the *Datapath Floating-Point Overview*. |

**Table 1-1 Pin Description**

| Pin Name | Width | Direction | Function |
|---|---|---|---|
| rnd | 3 bits | Input | Rounding mode<br>For details, see Rounding Modes in the *Datapath Floating-Point Overview*. |
| DG_ctrl | 1 bit | Input | Datapath gating control<br>■ 0: Component is disabled<br>■ 1: Normal component operation<br>For details, see "Datapath Gating Control with DG_ctrl" on page 3 |

**Table 1-2 Parameter Description**

| Parameter | Values | Description |
|---|---|---|
| sig_width | 2 to 253 bits | Word length of fraction field of floating-point numbers `a`, `b`, and `z` |
| exp_width | 3 to 31 bits | Word length of biased exponent of floating-point numbers `a`, `b`, and `z` |
| ieee_compliance | 0, 1, or 3<br>Default: 0 | Level of support for the IEEE Std 754 standards:<br>■ 0: No support for NaNs and denormals; NaNs are considered infinities and denormals are considered zeros<br>■ 1: Fully compliant with the IEEE Std 754-1985 standard, including support for NaNs and denormals<br>■ 2: Reserved<br>■ 3: Fully compliant with the IEEE Std 754-1985 standard plus NaN representation that matches the IEEE Std 754-2008 standard[a]<br>For details, see Compatibility with IEEE Std 754 Standards in the *Datapath Floating-Point Overview* |

a. Propagating payload information to the output during the NaN process, which is an optional feature specified in the IEEE Std 754-2008 standard, is not supported.

**Table 1-3 Synthesis Implementations**

| Implementation Name | Function | License Feature Required |
|---|---|---|
| rtl | Datapath gating close to the main inputs | ■ DesignWare (P-2019.03 and later)<br>■ DesignWare-LP (before P-2019.03) |
| rtl2[a] | Datapath gating to allow late arrival time of `DG_ctrl` signal | ■ DesignWare (P-2019.03 and later)<br>■ DesignWare-LP (before P-2019.03) |

a. By default, the rtl2 implementation is used for synthesis (for more, see "Datapath Gating Control with DG_ctrl" on page 3).

**Table 1-4 Simulation Models**

| Model | Function |
|---|---|
| DW02.DW_FP_ADDSUB_DG_CFG_SIM | Design unit name for VHDL simulation |
| dw/dw02/src/DW_fp_addsub_DG_sim.vhd | VHDL simulation model source code |
| dw/sim_ver/DW_fp_addsub_DG.v | Verilog simulation model source code |

**Table 1-5 Functional Description**

| a | b | op | status | z[a] |
|---|---|---|---|---|
| a (floating-point) | b (floating-point) | 0 | * | a + b (floating-point) |
| a (floating-point) | b (floating-point) | 1 | * | a - b (floating-point) |

a. The actual value of the result is defined by the rounding mode (`rnd`).

## Datapath Gating Control with DG_ctrl

For DW_fp_addsub_DG and other combinational components that have the datapath gating feature, the `DG_ctrl` port is provided to control datapath gating.

When `DG_ctrl` = 1, the component behaves as expected according to activity on the input ports.

When `DG_ctrl` = 0:

- The component is disabled and internal gates are totally or partially isolated to block propagation of switching activity inside the component. This makes the component less sensitive to switching activity on the main ports and reduces dynamic power consumption.
- Values at the output ports are not defined.
- Simulation models set 'X' values at the output ports.

The implementations for DW_fp_addsub_DG (see Table 1-3 on page 2) perform datapath gating differently:

- The rtl implementation places datapath gating as close as possible to the input ports to maximize dynamic power savings when `DG_ctrl` = 0. However, if the `DG_ctrl` signal arrives later than the data inputs, timing is degraded and area is increased to recover timing, which can increase power.
- The rtl2 implementation places datapath gating near the middle of the component. This approach is less sensitive to the arrival time of `DG_ctlr` and has a better chance of meeting timing and still providing dynamic power savings.

By default, the synthesis tool uses the rtl2 implementation, but you can override that. If timing constraints are loose or you know that the signal driving the `DG_ctrl` port arrives at the same time as other input signals, greater power savings can be attained by using the rtl implementation. You can make the override on a global level or on a case-by-case basis, as explained next.

To use the rtl implementation globally, you can disable the rtl2 implementation as follows:

- Design Compiler (before version P-2019.03):

```
set_dont_use {dw_minpower.sldb/DW_fp_div_DG/rtl2}
```

- Design Compiler (P-2019.03 and later)

```
set_dont_use {dw_foundation.sldb/DW_fp_div_DG/rtl2}
```

- Fusion Compiler:

```
set_synlib_dont_use {dw_foundation/DW_fp_div_DG/rtl2}
```

To use the rtl implementation for specific instantiated components, use the set_implementation command:

```
set_implementation U1 rtl
```

## Suppressing Warning Messages During Verilog Simulation

The Verilog simulation model includes macros that allow you to suppress warning messages during simulation.

To suppress all warning messages for all DWBB components, define the DW_SUPPRESS_WARN macro in either of the following ways:

- Specify the Verilog preprocessing macro in Verilog code:

```
`define DW_SUPPRESS_WARN
```

- Or, include a command line option to the simulator, such as:

`+define+DW_SUPPRESS_WARN` (which is used for the Synopsys VCS simulator)

The warning messages for this model include the following:

- If an invalid rounding mode has been detected on `rnd`, the following message is displayed:

```
WARNING: <instance_path>:
    at time = <timestamp>: Illegal rounding mode.
```

To suppress this message, use the DW_SUPPRESS_WARN macro explained earlier.

## Related Topics

- Datapath Floating-Point Overview
- DesignWare Building Blocks User Guide

## HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DWpackages.all;
use DWARE.DW_Foundation_comp.all;

entity DW_fp_addsub_DG_inst is
    generic (
      inst_sig_width : POSITIVE := 23;
      inst_exp_width : POSITIVE := 8;
      inst_ieee_compliance : INTEGER := 0
      );
    port (
      inst_a : in std_logic_vector(inst_sig_width+inst_exp_width downto 0);
      inst_b : in std_logic_vector(inst_sig_width+inst_exp_width downto 0);
      inst_rnd : in std_logic_vector(2 downto 0);
      inst_op : in std_logic;
      inst_DG_ctrl : in std_logic;
      z_inst : out std_logic_vector(inst_sig_width+inst_exp_width downto 0);
      status_inst : out std_logic_vector(7 downto 0)
      );
    end DW_fp_addsub_DG_inst;


architecture inst of DW_fp_addsub_DG_inst is

begin

    -- Instance of DW_fp_addsub_DG
    U1 : DW_fp_addsub_DG
    generic map ( sig_width => inst_sig_width, exp_width => inst_exp_width,
ieee_compliance => inst_ieee_compliance )
    port map ( a => inst_a, b => inst_b, rnd => inst_rnd, op => inst_op, DG_ctrl =>
inst_DG_ctrl, z => z_inst, status => status_inst );


end inst;
```

## HDL Usage Through Component Instantiation - Verilog

```
module DW_fp_addsub_DG_inst( inst_a, inst_b, inst_rnd, inst_op, inst_DG_ctrl,
          z_inst, status_inst );

parameter sig_width = 23;
parameter exp_width = 8;
parameter ieee_compliance = 0;


input [sig_width+exp_width : 0] inst_a;
input [sig_width+exp_width : 0] inst_b;
input [2 : 0] inst_rnd;
input inst_op;
input inst_DG_ctrl;
output [sig_width+exp_width : 0] z_inst;
output [7 : 0] status_inst;

    // Instance of DW_fp_addsub_DG
    DW_fp_addsub_DG #(sig_width, exp_width, ieee_compliance)
      U1 ( .a(inst_a), .b(inst_b), .rnd(inst_rnd), .op(inst_op),
.DG_ctrl(inst_DG_ctrl), .z(z_inst), .status(status_inst) );

endmodule
```

## Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

| Date | Release | Updates |
|---|---|---|
| October 2020 | DWBB_202009.1 | ■ For enhanced NaN compatibility with the IEEE Std 754 standards, added a new value for *ieee_compliance* in Table 1-2 on page 2 |
| July 2020 | DWBB_201912.5 | ■ Adjusted the description of the *ieee_compliance* parameter in Table 1-2 on page 2<br>■ Added "Suppressing Warning Messages During Verilog Simulation" on page 4 |
| March 2019 | DWBB_201903.0 | ■ Added "Datapath Gating Control with DG_ctrl" on page 3<br>■ Clarified some information about minPower<br>■ Added this Revision History table and the document links on this page |

# Copyright Notice and Proprietary Information



### Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

### Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

### Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

### Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

### Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com